LOCKSS:
Lots Of Copies Keep Stuff Safe

David S. H. Rosenthal

LOCKSS Program
Stanford University Libraries
http://www.lockss.org/

Overview

- Design
  - 3rd, 2nd party archives
  - Copyright
- Implementation
  - Prototype & lessons
  - Protocol research
- Deployment
  - Public network
  - Private networks
- Interoperability
  - Content
  - Metadata
  - Audit
- Lessons
  - Audit
  - Transparency
  - Licensing

Overview

- Design
  - 3rd, 2rd party archives
  - Copyright
- Implementation
  - Prototype & lessons
  - Protocol research
- Deployment
  - Public network
  - Private networks
- Interoperability
  - Content
  - Metadata
  - Audit
- Lessons
  - Audit
  - Transparency
  - Licensing

Design

- 3rd vs. 2nd party archives
  - Different legal context leads to different designs
- LOCKSS design principles
  - Minimize change to existing legal relationships
  - Reinstate the purchase model of paper
  - Preserve what the reader saw
  - Provide readers transparent access to content

Design Features

- Publisher adds permission to web site
  - Permission only visible to subscribers
  - Permission is preserved with the content
- Libraries collect content by crawling
  - Using their subscription access
- Preserved content is proxy for publisher
  - Delivered only if publisher refuses to deliver it
  - Doesn't deprive publisher of web hits
- Only library's readers can access its content
  - Minimize additional risk of content theft

Lots Of Copies

- One copy in each subscriber's LOCKSS box
  - 1 hour of lawyer = 5TB of disk
  - Abundance of copies changes the question
- How few copies to be as safe as needed?
  - Can't answer this
    - don't have necessary data or models
  - Answer often assumed to be 3
- What can be done to make the copies safer?
  - Can answer this
  - Use the abundance of copies to make each less critical

Preserve Against What?

Media failure

Hardware failure

Software failure

Network failure

Obsolescence

Natural Disaster

Preserve Against What?

Media failure

Hardware failure

Software failure

Network failure

Obsolescence

Natural Disaster

Operator error

External Attack

Insider Attack

Economic Failure

Organization Failure

Preserve How Well?

- Bit Preservation: A Solved Problem?
  - 1PB for 100yr with 50% chance of every bit undamaged
  - Each bit like radioactive atom that randomly decays
  - Specification: bit half-life > 60M times age of universe
- Can't know whether we meet this goal
  - Can't afford to do the experiments
- Can know we have improved
  - Have addressed threats that weren't addressed before

Overview

- Design
  - 3rd, 2nd party archives
  - Copyright
- Implementation
  - Prototype & lessons
  - Protocol research
- Deployment
  - Public network
  - Private networks
- Interoperability
  - Content
  - Metadata
  - Audit
- Lessons
  - Audit
  - Transparency
  - Licensing

Implementation

- Prototype (1998-2002)
  - Daemon process in Java
  - 6 then ~12 libraries, *Science* and *British Medical Journal*
- Lessons learned
  - System works, libraries can collect & preserve e-journals
  - Crawling with permission viable collection method
  - Needs plugin to handle publisher variability
  - Appliance architecture to reduce administration costs
  - Initial multicast protocol impractical

Research

- TCP-based V1 protocol vs. Black-Hat
  - Vulnerable to several attacks
  - So was improved V2
- Stanford CS research program
  - Assume open network, powerful enemy, no secrets
  - New techniques for P2P fault & attack tolerance
    - Best paper SOSP '03, ACM student research award
  - Also defense against DDoS attacks
    - Published at USENIX '05
- Basis for V3 (current) protocol

Overview

- Design
  - 3rd, 2nd party archives
  - Copyright
- Implementation
  - Prototype & lessons
  - Protocol research
- Deployment
  - Public network
  - Private networks
- Interoperability
  - Content
  - Metadata
  - Audit
- Lessons
  - Audit
  - Transparency
  - Licensing

Deployment

- Beta test 2002-2004
  - Total re-write of prototype, deployed to ~50 libraries
- Production 2004-
  - ~200 LOCKSS boxes in use, largest box 2TB content
- Private LOCKSS Networks (PLNs)
  - You-scratch-my-back-I'll-scratch yours organization
  - Many content genres
  - Cultural collections, GovDocs, state records, ETDs, …
  - Example: NDIIPP-funded MetaArchive

CLOCKSS PLN

- Community-governed low-cost dark archive
  - Libraries support, ~12 run archive nodes
  - 1st 7 nodes up: Japan, Australia, Canada, US
  - Publishers support, deposit content
  - Sustainability goal: endowment
- If content is no longer available
  - Board determines content has been triggered
  - Content extracted from archive then republished
  - Triggered content uses Creative Commons license
  - See examples at http://www.clockss.org/

Overview

- Design
  - 3rd, 2nd party archives
  - Copyright
- Implementation
  - Prototype & lessons
  - Protocol research
- Deployment
  - Public network
  - Private networks
- Interoperability
  - Content
  - Metadata
  - Audit
- Lessons
  - Audit
  - Transparency
  - Licensing

Content Interoperability

- Standard: WARC
  - Based on Internet Archive's ARC
- LOCKSS box export content as ARC files
  - Just as if Heritrix had crawled instead of LOCKSS
  - Special case of general re-crawl capability
- LOCKSS box import content as ARC files
  - Just as if LOCKSS had crawled instead of Heritrix
  - Special case of general packed format ingest facility
  - Used to replicate from Archive-It to LOCKSS PLN

Metadata Interoperability

- Format metadata – Standard: MIME
  - LOCKSS preserves HTTP headers, payload for all URLs
  - Transparent format migration uses MIME negotiation
- Bibliographic metadata – Standard: DC, DOI
  - LOCKSS boxes find article-level metadata in content
    - Plugin has publisher-specific code for this
  - Merge with journal-level metadata from Title DataBase
  - Use to serve content via OpenURL, DOI
  - Working with DNB, Humboldt – interoperate with KOPAL
    - Use METS to package metadata for exchange

Audit Interoperability

- Mutual audit
  - Is object in repository A same as in repository B?
  - LOCKSS protocol implements this
    - Simple, well-specified operations expressed in XML
  - For Web content this is tricky
    - Personalizations and other dynamic content
- 3rd Party audit
  - Does repository have every object it claims?
  - Is every object undamaged?
  - Without the auditor having access to the objects itself

Overview

- Design
  - 3rd, 2nd party archives
  - Copyright
- Implementation
  - Prototype & lessons
  - Protocol research
- Deployment
  - Public network
  - Private networks
- Interoperability
  - Content
  - Metadata
  - Audit
- Lessons
  - Audit
  - Transparency
  - Licensing

3rd Party Audit Requirements

- Don't trust repository being audited
- Don't transfer objects from repository to auditor
- Auditor should not be in the ingest pipeline
- Auditor should be fault-tolerant
- Audit checks should be combined with fixity checks

Lessons: Audit

- ISO-9000 style policy audit useful
  - But can't determine whether preservation is happening
- Existing techniques don't meet requirements
  - ACE: requires transfer of every object to auditor
    - Currently trusts repository being audited instead
  - Shah *et al*: requires 1/N transfer of objects to auditor
    - Doesn't trust repository being audited
- Important research topic
  - Archives cover up data loss incidents
  - Adversarial audits essential to deter cover up

Lessons: Transparency

- Transparent access to preserved content
  - Proxy provides this, but practical difficulties
  - Otherwise need to rewrite links – difficult & error-prone
- Transparency sounds good, but:
  - System needs to deliver visible value to readers
  - Otherwise hard to justify paying for it
- Memento proposal addresses this issue well
  - We're designing LOCKSS support for it

Lessons: Licensing

- Interoperability: legal not just technical issue
  - Even open access Web content is copyright
  - Need specific permission or use “safe harbor”
- Two big problems for successor archives
  - “Orphan Works” for preservation
  - What copyright law in force when interoperation needed?
- If at all possible use Creative Commons
  - Specific permission for all preservation activities

Conclusion

- Preservation is a security issue
  - Assuming a benign environment is a mistake
- Black Hat analysis is essential - examples:
  - LOCKSS V1 protocol, ACE
- Applies to interoperability standards too
  - Start from explicit threat model
  - Try to break the system for each threat in turn
  - Repository interoperating with may not be friendly